UNITED STATES PATENT AND TRADEMARK OFFICE

A

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/766,209	01/19/2001	Brandon J. Passanisi	P5505/14695.007001	9219

32615 7590 12/22/2005

OSHA LIANG L.L.P./SUN
1221 MCKINNEY, SUITE 2800
HOUSTON, TX 77010

EXAMINER	
VU, TUAN A	
ART UNIT	**PAPER NUMBER**
2193	

DATE MAILED: 12/22/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 09/766,209	Applicant(s) PASSANISI, BRANDON J.
	Examiner Tuan A. Vu	Art Unit 2193

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 November 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-6,8-11 and 13-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-6,8-11 and 13-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 11/14/2005.

As indicated in Applicant's response, claims 1, 9, 18 have been amended, with claim 20 canceled. Claims 1, 3-6, 8-11,13-18 are pending in the office action.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1, 3-6, 8-11, and 13-18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kobayashi, USPN: 6,633,888 (hereinafter Kobayashi) in view of admitted prior art (see Affidavit 1.132 filed 9/24/2004 and Anne Thomas' White paper -- hereinafter APA); and further in view of Template Software, inc, "WorkFlow Template - Using the WFT Development Environment", Copyright 1998 (hereinafter WFT) .

As per claim 1, Kobayashi discloses an apparatus for facilitating development of Java bundles, comprising:

a processor and a memory;

an integrated Development Environment to execute a module under the control of the processor to generate Java bundles (e.g. *visual builder 708* - Fig. 7; *JAR file* - step *808*, Fig. 8; step *1910*, Fig. 19; col. 7, lines 45-55 – Note: *Visual builder 708* reads on computer executing module for bean add-ons development purpose) using a plurality of development tools (*IDE 712* – Fig. 7; col. 11, lines 26-30 – Note: a IDE reads on plurality of tools), wherein the tools for comprise a manifest generator tool for creating a manifest file for the bundles (e.g. creator 308 – Fig. 3; step *808* –Fig. 8).

But Kobayashi does not disclose that the Java bundles are Java Embedded Server bundles nor the manifest file being JES manifest file. The use of Java compacted packages like JAR in network distribution to facilitate their deployment at target devices where storage resources are limited, e.g. resources-restraint embedded devices where was a known concept at the time the invention was made. This concept is evidenced via embedded systems with capabilities to install and deploy portable package of software such as the likes Java Embedded Servers as taught by APA (see Affidavit 1.132 filed 9/24/2004 and Anne Thomas' White paper). Based on Kobayashi's suggestion that cross-platform Java packages (or JAR files) or bytecodes can be deployed on embedded processing units (e.g. step *808*, Fig. 8; col. 7, lines 19-30) it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the Java package builder method by Kobayashi so that the Java bundles along with their manifest file are created using a IDE as mentioned above for facilitating the user-driven and dynamic creation and deployment at embedded systems in general and in JES in particular as by APA because, by bundling Java deployable components in small devices (or dedicated servers like JES) whose resources can be a limiting factor, the embedded systems (e.g. JES as one intended use) storage resources for otherwise non-bundled components would be alleviated, and also because according to APA, this IDE building application would support dynamic and as-needed application deployment (i.e. on-demand applications) on embedded systems utilizing, among other benefits, the known transportability of Java bytecodes (as mentioned by Kobayashi col. 7, line 46 to col. 8, line 7), whose implicit cross-platform portability can improve deployment of program components over the network as shown by JES/APA, thus making this remote deployment independent of any single server (Koyabashi, col. 8, top; see APA, On-Demand Java Applications, pg. 1).

Kobayashi (in view of APA) discloses generating the manifest file for the JES bundles; but Kobayashi does not explicitly disclose that the JES manifest generator is configured to: include one manifest header name associated with one manifest header by checking a box next to the header name; enter the value of one manifest header into a text field beside the manifest header name; select a name for the manifest file. Activating a button/box or clicking on an entry from a scroll-down selection in order to populate an adjacent text field (associated with such selection) was disclosed in many visual editors or authoring tools like the builder of Kobayashi at the time the invention was made; and WFT, in a development visual tool to create and edit software components similar to the endeavor of Kobayashi's visual builder (in view of APA) discloses a name selector next to a text field (see Fig. 2-8, ...*grayed out to indicate* - pg. 2-36; pg. 2-7→2-9) and further teaches a text field that changes color requiring thereby that the developer must enter a text entry thereinto. It would have been obvious for a visual tool like that of Kobayashi (in view of the JES manifest file creation by APA) to include GUI options (in Kobayashi's creation of a manifest file) for creating a file header to each manifest file using the text field for (i) manual entry – entering a value -- and (ii) for automated entry via mouse click selection – checking a box -- as mentioned above by WFT because each manifest file is known to be used in conjunction with the deploying of a package which the very manifest describes; and imparting a header name to the manifest would facilitate the reconstruction of the content of the package via mapping with the manifest file whose header name is determined as taught above by

WFT in the use of a manifest file as intended by Koyabashi – to correspond to one JES bundle (in view of APA) when the bundle is developed.

As per claim 3, Kobayashi discloses a window menu within a visual builder to retrieve components to assemble a Java bundle in the IDE (e.g. Fig. 7; Fig. 13-17; step *1910* – Fig. 19; col. 11, lines 26-30); hence has disclosed implicitly disclosed drop-down menu for accessing the module as to create Java bundles.

As per claim 4, Kobayashi discloses a update mechanism in the IDE (e.g. EDIT – Fig. 11; Fig. 13-17; step *2014* – Fig. 20).

As per claim 5, Kobayashi discloses a code template (e.g. Fig. 4; Constructor bean 1000, method bean 1012 – Fig. 10 – Note: a container for Java code methods and attributes is equivalent to class templates containing elements from which to construct further code classes)

As per claim 6, Kobayashi discloses a interface template (e.g. *palette ... internal interface* - col. 17, line 62 to col. 18, 8; col. 11, lines 45-49; *environment add-on 700*– Fig. 7; Fig. 14, 17); and implementation template (e.g. *visual builder 214* – Fig. 2; transport *API* 206 – Fig. 2; Fig. 9—Note: using a visual palette to effect APIs and interface calls to constructs classes and methods for beans reads on implementation template) but only teaches a activator functionality based on events (e.g. Fig. 16, 18-20 – Note: visual display to applying binding and linking properties of created components and to test beans reads on activator template). Given the visual aspect of activating the created components based on the builder template thus suggested by Kobayashi, it would have been obvious for one of ordinary skill in the art at the time the invention was made to add to the visual menu-driven tool of Kobayashi a activator template with which the components can be activated or linked just as suggested above because providing all the event-based functionalities for test or dynamic binding in one such graphical container module, i.e. a template, would provide the integration tool with one differentiated graphical module encompassing all the debugging and testing functions typical of the post-implementation stages of development as suggested above prior to delivering the package as built.

As per claim 8, Kobayashi (in view of APA) discloses a Java Embedded Server (JES) jar packager tool that packages JES bundles (e.g. step 1910 – Fig. 19).

As per claim 9, Kobayashi does not disclose link to JES-related web pages but teaches a Java bean-compliant visual tool of the likes of Visual Age *WebRunner* for a world wide web with Corba connectivity (e.g. *CORBA* - col. 5, lines 23-31; col. 11, lines 8-17; col. 23, lines 22-34). The creation of links to remote network storages is implied or explicit in all visual development tool (e.g. Kobayashi: Fig. 3-9 – Note: for example, CORBA and RMI retrieval of remote beans/reuse component in a Web-based connection is implied that web page link, e.g. URL, are implicit) or in JES/APA when web and browser application-based tools (with inherent web protocol connectivity features) are taught (see APA White Paper). Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to create the visual builder by Kobayashi so that using a JES as mentioned in claim 1, JES-related links to remote storage or URL-directed pages are effected via browser pages are effected in the builder because these would enable retrieval of components supporting the IDE development at the JES as taught by APA and initially desired by the bean-compliant visual builder by Kobayashi.

As per claim 10, this claim recites code template, manifest generator tool, and jar packager tool. All of which limitations have been addressed in claims, 5, 7, and 8 respectively.

As per claim 11, Kobayashi (in combination with APA) discloses a method comprising a processor, a memory as well as software instructions stored therein as recited in the apparatus claim 1, including combining a plurality of development tools in a module for the creation of Java bundles and execute the module in an IDE (e.g. *visual builder 708* - Fig. 7; Fig. 8; step *1910*, Fig. 19; col. 7, lines 45-55 – Note: *Visual builder 708* reads on module for bean add-ons and development modules being integrated in the IDE environment , *IDE 712*) wherein the tools comprise a manifest generator tool for creating a manifest file for the bundles (e.g. creator 308 – Fig. 3; step *808* –Fig. 8).

But Kobayashi does not specifically disclose that the bundles are Java Embedded Server bundles and that the manifest files are JES manifest files. This limitation has been however addressed in claim 1 above.

Nor does Kobayashi disclose that the JES manifest generator is configured to: include one manifest header name associated with one manifest header by checking a box next to the header name; enter the value of one manifest header into a text field beside the manifest header name; select a name for the manifest file. But in view of WFT, this limitation has been addressed in claim 1 above.

As per claim 13, Kobayashi discloses code samples (refer to rejection of claim 5).

As per claims 14 and 15, these claims correspond to the limitations of claims 7 and 8, hence are rejected with the corresponding rejection as set forth therein respectively.

As per claim 16, refer to rejection of claim 9.

As per claim 17, refer to claim 10.

As per claim 18, Kobayashi discloses an apparatus for facilitating development of Java bundles, comprising means for:

providing sample code segments (refer to rejection of claim 5);

creating Java manifest files for the bundles (re to rejection of claim 1); and

packaging the bundles (re claim 8);

executing a module into an Integrated Development Environment (e.g. *IDE 712* – Fig. 7; col. 11, lines 26-30), the module comprising a plurality of development tools (e.g. Fig. 7 - Note: *Visual builder 708* reads on module for bean add-ons and development modules being integrated in the IDE environment , *IDE 712*); wherein the tools executing in the IDE comprise a manifest generator tool for creating a manifest file for the bundles (e.g. creator 308 – Fig. 3; step *808* – Fig. 8).

But Kobayashi does not specifically disclose that the Java bundles are Java Embedded Server bundles and that the manifest files are JES manifest files. These limitations have been however addressed in claim 1 above.

Nor does Kobayashi disclose that the JES manifest generator is configured to: include one manifest header name associated with one manifest header by checking a box next to the header name; enter the value of one manifest header into a text field beside the manifest header name; select a name for the manifest file. But in view of WFT, this limitation has been addressed in claim 1 above.

Response to Arguments

4. Applicant's arguments filed 11/14/2005 have been fully considered but they are not persuasive in some part and moot in other parts. Following are the observations in regard thereto.

(A) Applicant has submitted that neither Koyabashi does not teach checking a box next to a list of manifest header … manifest files' (Appl. Rmrks, pg. 7, bottom); but the newly added limitations are now addressed with a new grounds of rejection, rendering the arguments thereby moot.

(B) Applicant has submitted that APA is silent about disclosing a manifest generator tool, that there is not any suggestion or motivation to combine Koyabashi and APA for generation of JES bundles, that one skill in the art would not find including a manifest generator tool in a IDE to be obvious (Appl. Rmrks, pg. 8, 2nd and middle para). In response, it is noted that the VEA's tool of Koyabashi combines a plurality of tools into a visual add-on tool wherein the IDE as well as a bean compiler are also disclosed to be part of such builder environment (see col. 8, lines 20-65; col. 11, lines 7-30). The IDE and the compiler means - to bundle beans and to include the manifest creator -- are integral to the overall visual builder (see Koyabashi: step *808* – Fig. 8), making what amounts to an Integrated Development Environment to read exactly on what Koyabashi's visual builder amounts to not only creating the Java bundles but to associate a manifest file therewith. While it is known that a manifest file is for providing information as to the contents of a distributed package that such package would be deployed, such dependency of the package content to the information of the manifest file is evidenced in Koyabashi's Fig. 8; and the Manifest being associated with the JAR by APA (see ch. 4- *ServiceSpace … Deployment Framework*). One skill in the art of when faced with teaching as to distributing of beans like via Koyabashi's builder framework to encapsulate delivered beans in package with information coming from a manifest file, would find the manifest file teaching by APA very much in the same distribution context to devices for facilitating software redeployment in remote systems independent of a single server (see Koyabashi, col. 8, top); and this benefit can be perceived in APA mentioning of the extended capability of such package *without the need of another device* (see APA, On-Demand Java Applications, pg. 1). There is evidence leading to the rationale as to why there is need to create a manifest file in conjunction to building packages for a remote device software demand in view of such extensibility and easy to redeployed packages owing to the information from a manifest file. The arguments about lack of suggestion or motivation to combine would amount to mere allegations for they appear to have been construed without capturing the very endeavor of each references (Koyabashi's package or JAR-manifest and APA's manifest-bundle) based on which a purpose or a common need has been detected, such need leading to a motivation as so set forth in the rejection. Besides, the claim does not provide specific teaching as that would enable the implementation of a JES packaging -- for the very need of a server environment -- would necessary preclude a visual builder for creating beans archive or packaging thereof as from Koyabashi from reading on a JES bundling for a server environment --such as APA is propounding-- in light of the on-demand applications on remote devices and of code deployment in platform independent format and the extensibility of the packages used therefor, as perceived for one skill in the art at the time the invention was made (see section B).

(C) Applicant has submitted that amended claims 1, 11 and 18 are patentable because neither reference has been able, separately or jointly, to render the claims obvious. For lack of specificity in the claimed subject matter in some extent and in combination with the above explanations in section B above in other extent, first, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, there is evidence from the references that redeployment can benefit of the way the bundles are created to facilitate the use of package content independent of a outside controller. Second, the Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

For the above reasons, the claims will stand rejected as set forth above.

Conclusion

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

Tuan A Vu

Patent Examiner,
Art Unit 2193
December 20, 2005